

Actividad | 3 | Programa

Banco Mexicano (Parte 2)

Lenguajes de Programación IV

Ingeniería en Desarrollo de Software

TUTOR: Aarón Iván Salazar Macías

ALUMNO: Adriana Esteban López

FECHA: 09 de Agosto de 2025

INDICE

INTRODUCCIÓN	03
DESCRIPCIÓN	04
JUSTIFICACIÓN	06
DESARROLLO	07
CONCLUSIÓN	18
REFERENCIAS	19

INTRODUCCIÓN

NetBeans es un entorno de programación que es utilizado para programar en diferentes lenguajes, entre ellos Java.

Las principales ventajas de NetBeans son:

- **Código abierto:** Lo cual indica que es gratuito.
- **Multiplataforma:** Se puede emplear en distintos dispositivos, así como ejecutarse en diferentes sistemas operativos.
- **Manejo automático de la memoria:** Para aquellos programas con C o C++, la administración de la memoria se puede realizar de forma automática.
- **Multilenguaje:** es decir, no solo opera con Java; sino que NetBeans puede utilizar otros lenguajes como PHP, C o Ruby, entre otros.

Como ya se mencionó, uno de los lenguajes de programación más utilizados en NetBeans es Java, el cual es un lenguaje orientado a objetos, lo cual implica que se deben de crear clases; así mismo dentro del desarrollo de esta actividad, estaremos haciendo uso de la Programación Orientada a Objetos (POO) para la interfaz que el usuario estará visualizando al momento de utilizar el sistema; dentro de la POO cada uno de los objetos es una entidad que tiene unas propiedades particulares, los atributos y las formas de operar de ellos los métodos.

NetBeans facilita la POO en Java debido a:

- **Editor de código,** el cual permite la creación de clases, objetos, atributos y métodos de manera eficiente.
- **Compilador,** el cual traduce el código Java a código de máquina que la computadora puede ejecutar.
- **Asistentes y plantillas,** el cual ayuda a la generación de código de manera automática

El desarrollo de esta actividad requiere también de la realización de una pequeña Base Datos, la cual estaremos desarrollando a través de XAMPP y MySQL.

DESCRIPCIÓN

Contextualización:

Los clientes de Banco Mexicano necesitan un programa que les permita a sus clientes el realizar depósitos, retiros y consultas de su saldo. Por lo que necesitan que un ingeniero en desarrollo de software genere una base de datos que atienda a esta necesidad.

Actividad:

Utilizar el lenguaje Java 8 y el entorno de programación sugerido en la sección de Recursos para realizar un programa con los siguientes requerimientos:

La pantalla principal debe contar con un menú que tenga las siguientes opciones. Además, deberá solicitar la respuesta por teclado:

1. Depósito
2. Retiro
3. Saldo
4. Salir

Respuesta: En caso de que el usuario ingrese la opción Depósito, el programa deberá ser capaz de capturar por teclado los siguientes datos:

- Cantidad a depositar
- Preguntar si desea realizar otro depósito; en caso de que se seleccione la opción “Si”, deberá mostrar nuevamente la pantalla de Depósito, si la respuesta es “No”, deberá mandar al menú principal

En caso de que el usuario ingrese la opción Retiro, el programa deberá ser capaz de capturar por teclado los siguientes datos:

- Cantidad a retirar.
- Posteriormente deberá de mandar al menú principal.

Es necesario que el programa esté realizado completamente en el paradigma de la programación orientada a objetos, y cumplir con los principios de esta.

Ejemplo de estructuras a utilizar:

- Para las validaciones, utilizar sentencias IF, FOR, Switch-case entre otras, según sea conveniente.

Ahora dentro del desarrollo de esta actividad, se estará realizando la conexión que se requiere con la Base de Datos para estar ejecutando cada una de las opciones del menú principal de manera correcta, es decir, estaremos realizando la programación de los eventos de las diferentes opciones que el usuario puede elegir, Depositar, Retirar y Saldo

JUSTIFICACIÓN

Como ya se menciona dentro de la Introducción, las principales ventajas de NetBeans:

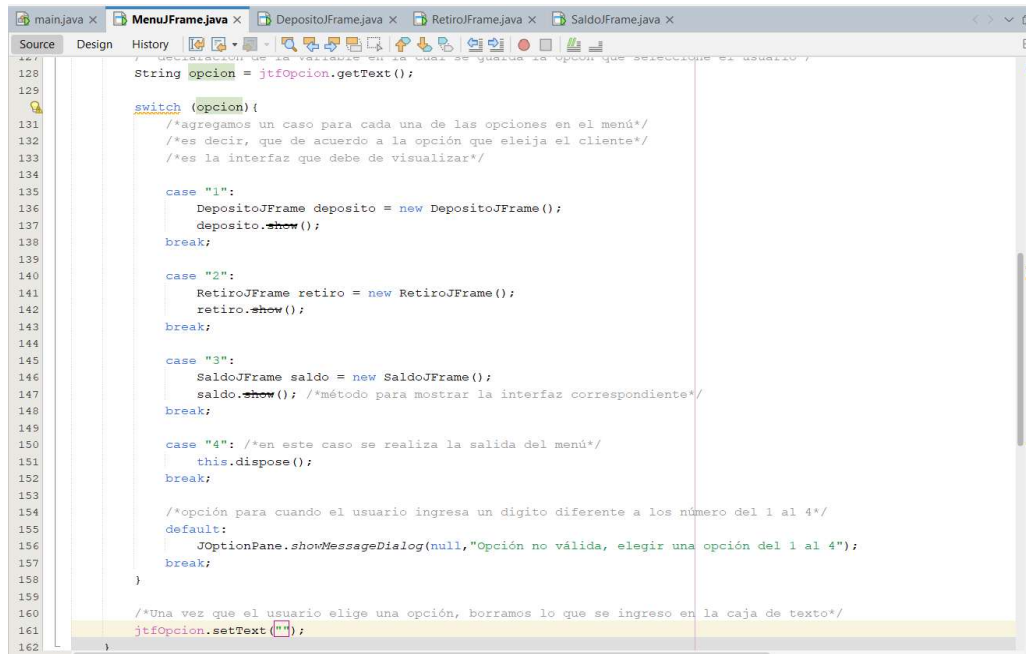
- **Ajustes de la interfaz de usuario** a través de la gestión de menús y barras de herramientas.
- **Establecimiento de la configuración de usuario.**
- **Gestión de almacenamiento** para guardar datos, así como para realizar su respectiva carga.
- **Gestión de ventana.**
- **Librería visual** que permite adquirir, por ejemplo, distintos **widgets**.
- **Recursos de desarrollo integrado**, como puede ser un **editor de texto fuente de Netbeans**.

Así mismo, el programar en Java también tiene sus ventajas, ya que es un lenguaje de programación completamente orientada a objetos, además de que también es de uso gratuito y portable.

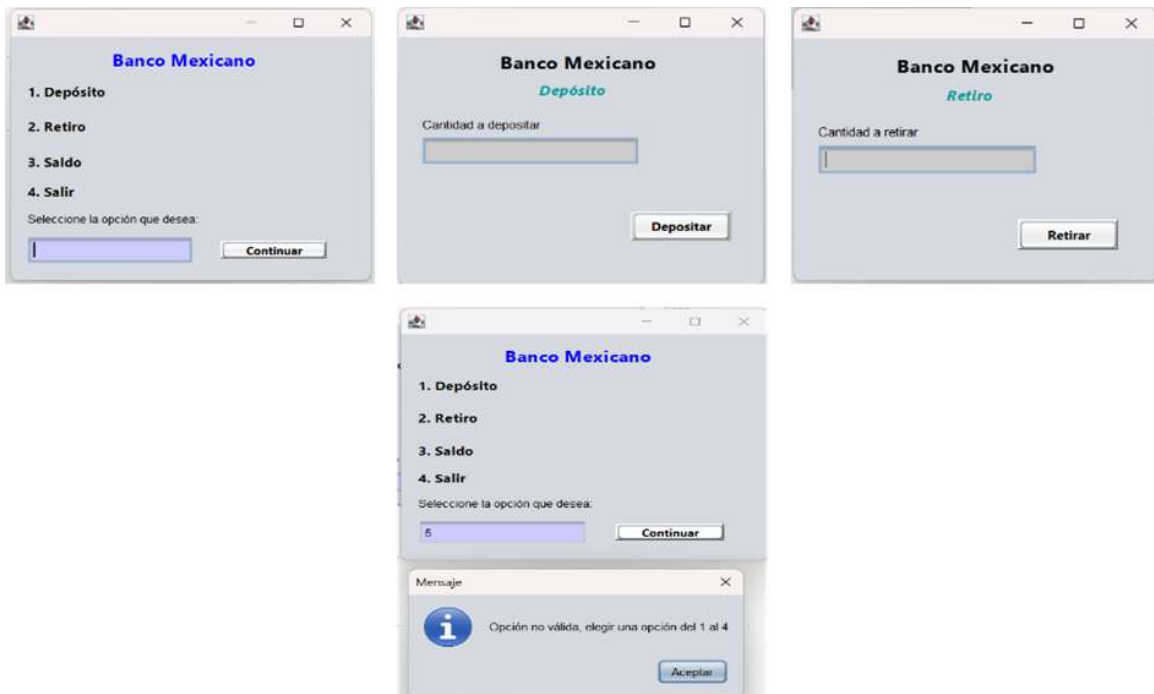
Por otro lado el uso de XAMPP y MySQL juntos es de gran utilidad, ya que XAMPP permite tener un entorno que facilita la instalación y configuración de MySQL, este último de fácil uso también para el programador, ya que las bases de datos pueden hacerse tanto con el uso de las herramientas visuales que proporciona como con código.

DESARROLLO

En el desarrollo de la actividad No. 2 lo que realizamos fue la programación del botón Continuar del menú principal para que mientras el usuario este eligiendo una opción del 1 al 4, ingrese en la aplicación correspondiente según la elección del usuario.



```
128 String opcion = jTextFieldOption.getText();
129
130 switch (opcion) {
131     /*agregamos un caso para cada una de las opciones en el menú*/
132     /*es decir, que de acuerdo a la opción que elija el cliente*/
133     /*es la interfaz que debe de visualizar*/
134
135     case "1":
136         DepositoJFrame deposito = new DepositoJFrame();
137         deposito.show();
138         break;
139
140     case "2":
141         RetiroJFrame retiro = new RetiroJFrame();
142         retiro.show();
143         break;
144
145     case "3":
146         SaldoJFrame saldo = new SaldoJFrame();
147         saldo.show(); /*método para mostrar la interfaz correspondiente*/
148         break;
149
150     case "4": /*en este caso se realiza la salida del menú*/
151         this.dispose();
152         break;
153
154     /*opción para cuando el usuario ingresa un digito diferente a los número del 1 al 4*/
155     default:
156         JOptionPane.showMessageDialog(null, "Opción no válida, elegir una opción del 1 al 4");
157         break;
158 }
159
160 /*Una vez que el usuario elige una opción, borramos lo que se ingreso en la caja de texto*/
161 jTextFieldOption.setText("");
162 }
```



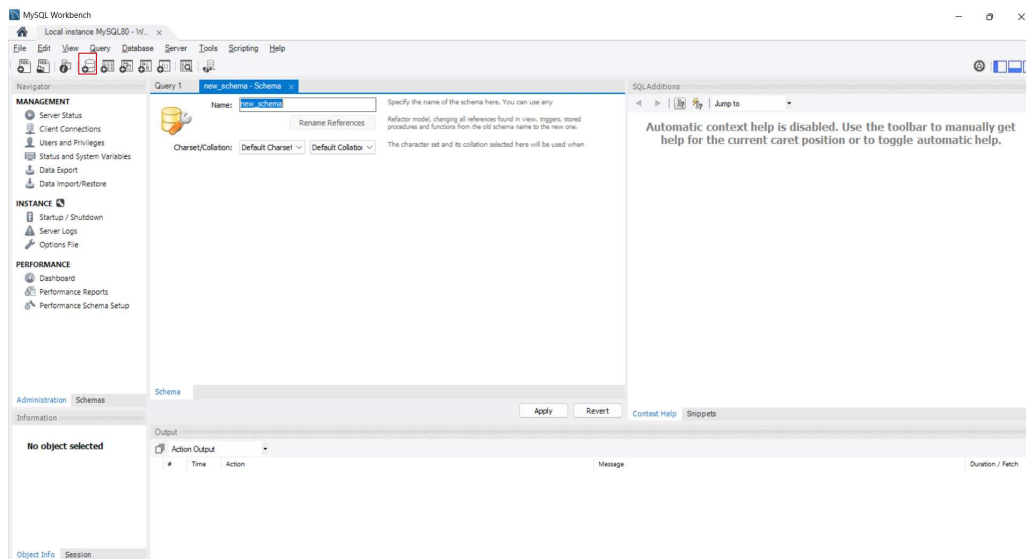
A partir de aquí estaremos realizando la conexión con la Base de Datos con la creación de la Clase *Connection*:

```

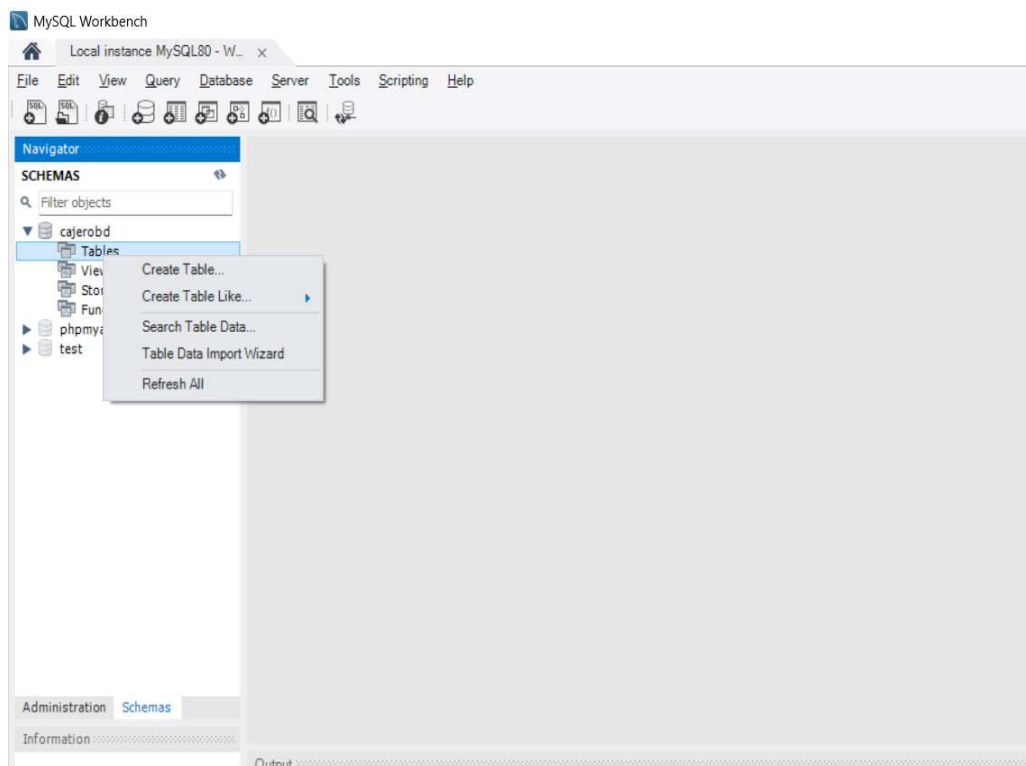
main.java X  MenuFrame.java X  DepositoFrame.java X  RetiroFrame.java X  SaldoFrame.java X  Conexión.java X
Source  History  [Icons]
8      import java.sql.DriverManager; /*Se agrega esta libreria para la línea de código 28*/
9
10     /**
11      *
12      * @author adri2
13      */
14     public class Conexión {
15         /*Esta Clase sirve para conectar la Base de Datos con la aplicación que el usuario estará utilizando*/
16
17         /*Definición del método Conexión*/
18         public Connection getConnection(){
19             Connection con = null;
20             String base = "cajerobd";
21             String url = "jdbc:mysql://localhost:3306/" + base;
22             String user = "root";
23             String password = "";
24
25             try {
26                 Class.forName("com.mysql.jdbc.Driver");
27                 con = (Connection) DriverManager.getConnection(url, user, password);
28
29             } catch (Exception e){
30
31                 System.err.print(e);
32
33             }
34
35             return con;
36         }
37     }
38
39
40

```


Ahora vamos a crear la Base de Datos, para lo cual estaremos haciendo uso de XAMPP y MySQL.



Damos clic en el ícono que esta marcado con el recuadro rojo, y daremos el nombre a la base de datos, que en este caso es **cajerobd** (es el nombre que utilizamos en la clase que ya creamos en NetBeans) y damos click en aplicar hasta finalizar.



Podemos observar que de lado izquierdo nos aparece el nombre de la Base de Datos que acabamos de crear, desplegamos el menú y en **Tables**, elegimos la opción de **Create Table** para crear las siguientes tablas: cuenta

Table Name: Schema: **cajerobd**

Charset/Collation: Default Collation: Engine:

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
saldo	FLOAT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Column Name: Data Type:

Charset/Collation: Default Collation:

Comments:

Storage: ☐ Virtual ☐ Stored

☐ Primary Key ☐ Not Null ☐ Unique

☐ Binary ☐ Unsigned ☐ Zero Fill

☐ Auto Increment ☐ Generated

Columns Indexes Foreign Keys Triggers Partitioning Options

Apply Revert

Una vez que se coloca el nombre de la Tabla y sus atributos, damos clic en Aplicar, en la parte inferior podemos ir visualizando si las acciones que vamos ejecutando se están realizando de manera correcta, porque estarán marcadas con un ícono característico:

MySQL Workbench

Local instance MySQL80 - W...

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

cajerobd

Tables

Views

Stored Procedures

Functions

phoenixadmin

test

Administration Schemas

Information

Schema: **cajerobd**

Object Info Session

Limit to 1000 rows

1 * `SELECT * FROM cajerobd.cuenta;`

Result Grid

id	saldo
1	1000

Output

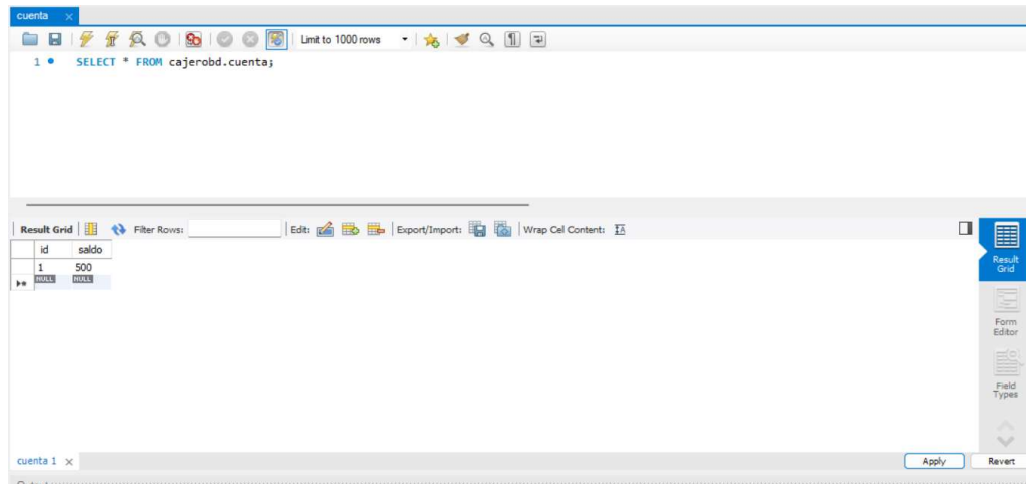
Output

#	Time	Action	Message	Duration / Fetch
1	00:01:50	Apply changes to cajerobd	Changes applied	
2	00:11:55	Apply changes to cuenta	Changes applied	
3	00:13:22	SELECT * FROM cajerobd.cuenta LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec

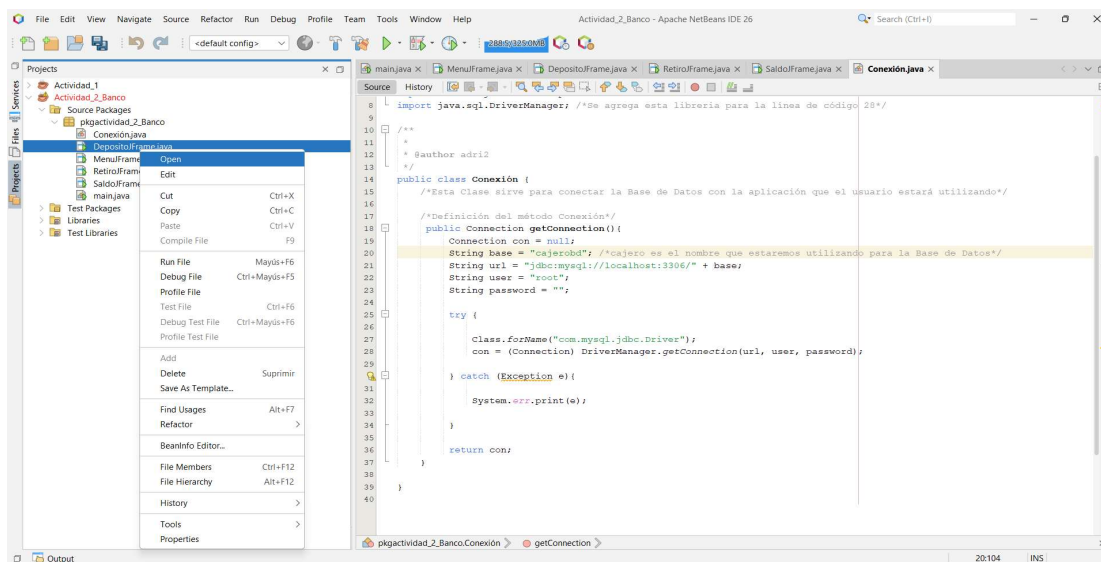
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Skipsets

Para ingresar datos en la tabla en sus diferentes campos, del lado izquierdo desplegamos el menú de **Tables** y en cada tabla seleccionamos el ícono de tabla para empezar a ingresar los datos y una vez que damos clic en aplicar, estos se estarán guardando en la tabla:



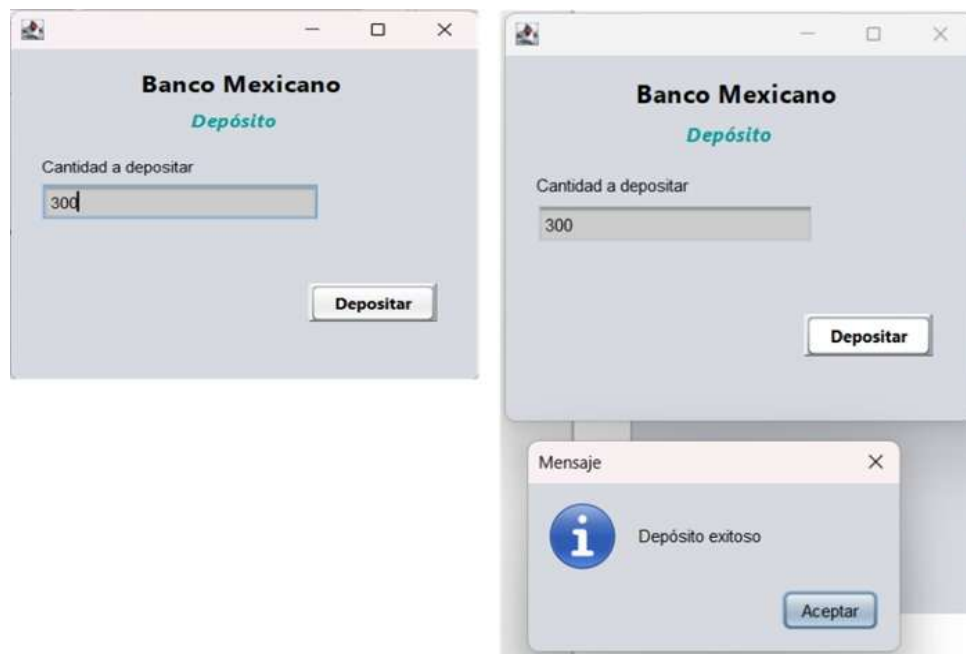
Ahora en NetBeans vamos a iniciar con la programación de la opción **DEPOSITO**:

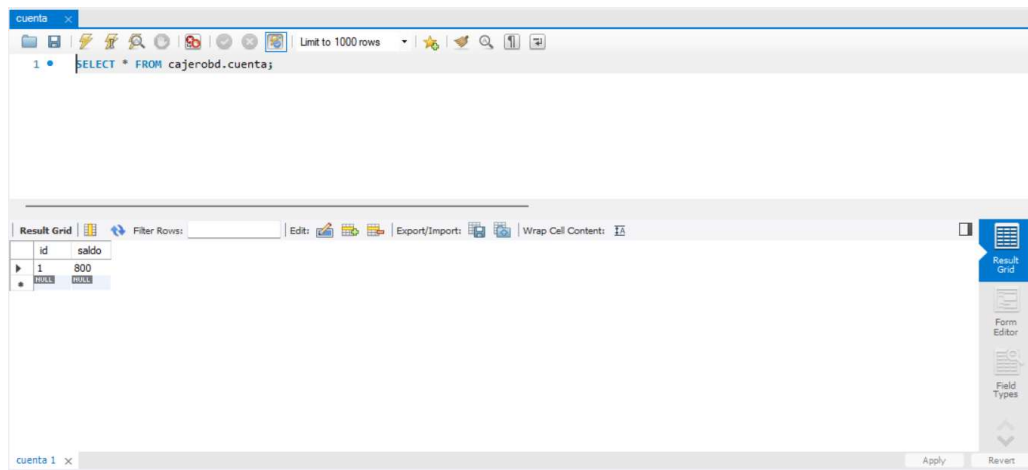


Seleccionamos el Frame de Deposito, damos clic derecho y seleccionamos la opción de **OPEN** para abrir el Frame y dar doble clic en el botón de **DEPOSITAR** para generar el evento:

```
main.java x MenuFrame.java x DepositoJFrame.java x RetiroJFrame.java x SaldoJFrame.java x Conexion.java x
Source Design History
28
29 /**
30  * This method is called from within the constructor to initialize the form.
31  * WARNING: Do NOT modify this code. The content of this method is always
32  * regenerated by the Form Editor.
33  */
34 @SuppressWarnings("unchecked")
35 Generated Code
112
113 private void jButtonDepositarActionPerformed(java.awt.event.ActionEvent evt) {
114     // TODO add your handling code here:
115     Connection con;
116     Conexion conn = new Conexion();
117     try {
118         con = conn.getConnection();
119         /*Realiza la consulta a la tabla cuenta*/
120         PreparedStatement ps = con.prepareStatement("UPDATE cuenta SET saldo = saldo + ? WHERE id=1");
121         /*con el signo de ? le estamos indicando que le sume lo de la caja de texto al campo saldo en la tabla cuenta*/
122         ps.setString(1, jTextFieldCantidad.getText());
123         /*ejecución de la consulta del UPDATE*/
124         int res = ps.executeUpdate();
125         if (res > 0) {
126             JOptionPane.showMessageDialog(null, "Depósito exitoso");
127         } else {
128             JOptionPane.showMessageDialog(null, "Error en depósito");
129         }
130         con.close(); /*cierre de conexión*/
131     } catch (Exception e) {
132         /*Si hay una error, esta línea nos muestra cuál es el error*/
133         System.err.println(e);
134     }
135 }
136
137 /**
138  * @param args the command line arguments
```

Dentro de este evento lo que hacemos es entablar conexión con la base de datos para que al momento de que el usuario ingrese el monto a depositar, este sea agregado en la tabla y campo correspondiente, veamos la ejecución:





Podemos observar que ahora tenemos ya los \$800.00.

Pasamos ahora al desarrollo de la opción de **RETIRO**, para lo cual en el Frame Retiro vamos a dar doble clic en el botón de RETIRAR para generar el evento correspondiente.

```

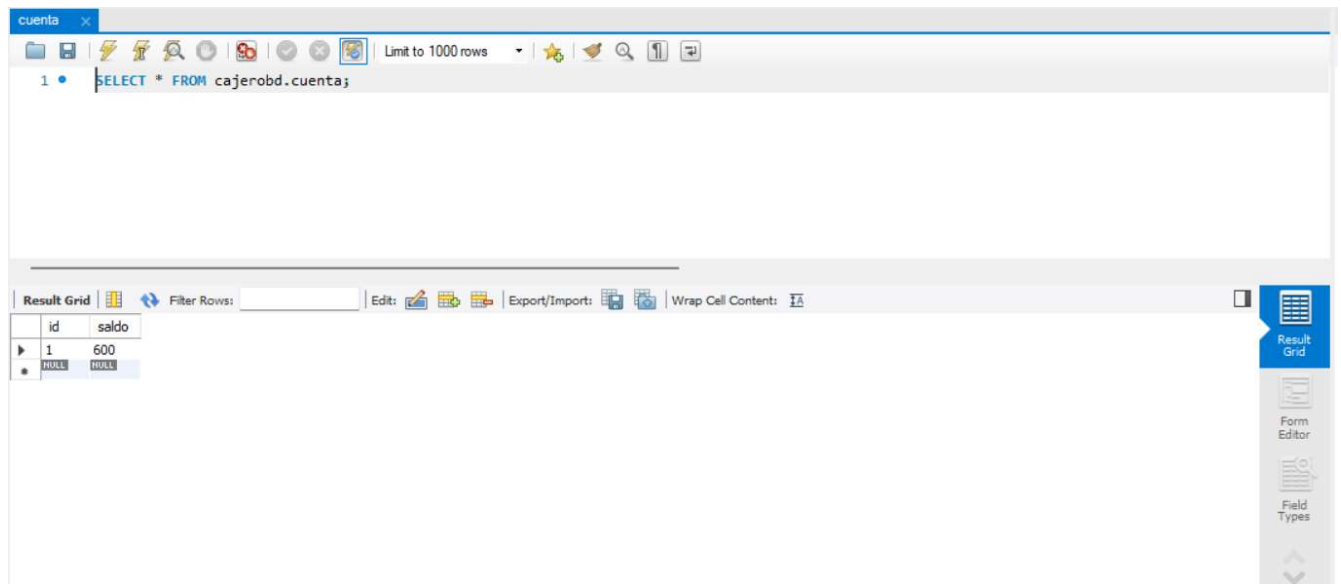
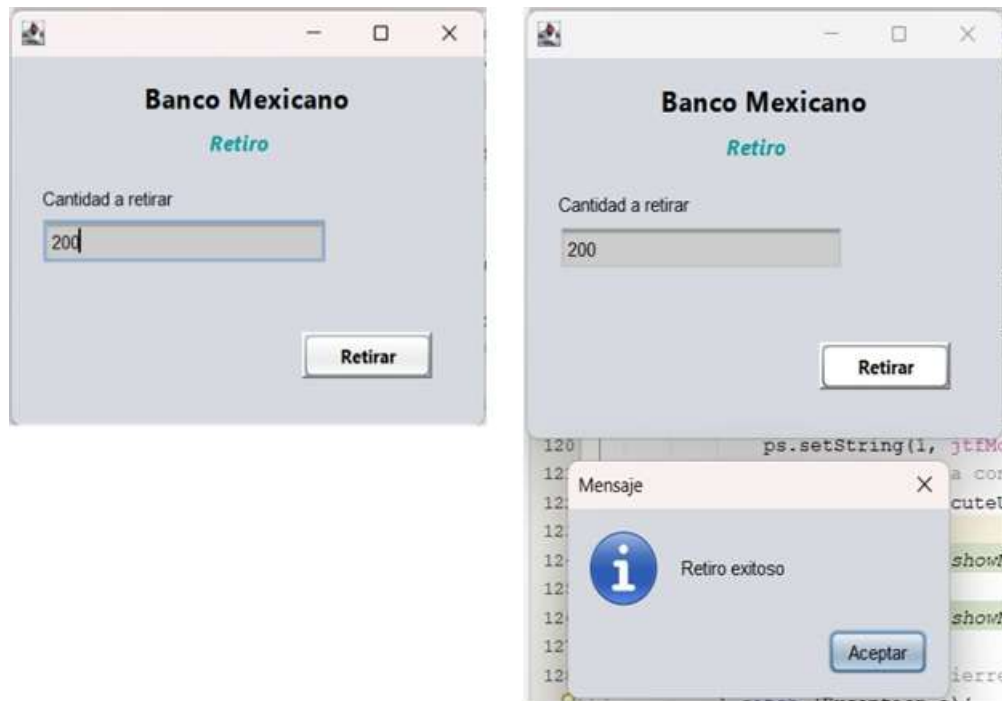
31  @SuppressWarnings("unchecked")
32  Generated Code
109
110  private void jButtonRetiroActionPerformed(java.awt.event.ActionEvent evt) {
111      // TODO add your handling code here:
112
113      Connection con;
114      Conexion conn = new Conexion();
115      try {
116          con = conn.getConnection();
117          /*Realiza la consulta a la tabla cuenta*/
118          PreparedStatement ps = con.prepareStatement("UPDATE cuenta SET saldo = saldo - ? WHERE id=1");
119          /*con el signo de ? le estamos indicando que le reste lo de la caja de texto al campo saldo en la tabla cuen
120          ps.setString(1, jTextFieldMonto.getText());
121          /*ejecución de la consulta del UPDATE*/
122          int res = ps.executeUpdate();
123          if (res > 0) {
124              JOptionPane.showMessageDialog(null, "Retiro exitoso");
125          } else {
126              JOptionPane.showMessageDialog(null, "Error en retiro");
127          }
128          con.close(); /*cierre de conexión*/
129      } catch (Exception e) {
130          /*Si hay una error, esta línea nos muestra cuál es el error*/
131          System.err.println(e);
132      }
133  }
134
135  /**
136   * @param args the command line arguments
137   */
138  public static void main(String args[]) {
139      /* Set the Nimbus look and feel */
140      Look and feel setting code (optional)
141  }
142
143  }
144
145  }
146
147  }
148
149  }
150
151  }
152
153  }
154
155  }

```

Básicamente podemos observar que es el mismo código que se utilizó en el botón de **DEPOSITAR** que tenemos en el Frame de DEPOSITO, solo que hacemos la observación de que en la siguiente línea de texto cambiamos el signo + por el signo -

PreparedStatement ps = con.prepareStatement("UPDATE cuenta SET saldo = saldo - ? WHERE id=1");

Vamos a realizar la ejecución, recordemos que actualmente el saldo que tenemos es de \$800, para este ejemplo, vamos a retirar un monto de \$200 y por tanto nos deben de quedar \$600.



Realizamos la misma acción de dar doble click en el botón **CONTINUAR** del FramSaldo, para la programación del evento correspondiente.

Una vez que se genera el evento correspondiente, nos colocamos en la parte del código del constructor para agregar la siguiente codificación:

```

16 public class SaldoJFrame extends javax.swing.JFrame {
17
18     private static final java.util.logging.Logger logger = java.util.logging.Logger.getLogger(SaldoJFrame.class.getName());
19
20     /**
21      * Creates new form SaldoJFrame
22      */
23     public SaldoJFrame() {
24         initComponents();
25
26         Connection con;
27         Conexion conn = new Conexion();
28         try {
29             con = conn.getConnection();
30             /*Realiza la consulta a la tabla cuenta*/
31             PreparedStatement ps = con.prepareStatement("SELECT saldo FROM cuenta WHERE id=1");
32             ResultSet rs = ps.executeQuery();
33
34             /*con este if, estamos indicando que si en la variable rs tenemos un valor, lo muestre*/
35             if (rs.next()) {
36
37                 jtfCantidad.setText(rs.getString("saldo"));
38
39             } else {
40
41                 JOptionPane.showMessageDialog(null, "Error");
42
43             }
44
45             con.close(); /*cierre de conexión*/
46         } catch (Exception e) {
47             /*Si hay una error, esta línea nos muestra cuál es el error*/
48             System.err.println(e);
49         }
50     }

```

Al momento de ejecutar este código lo que nos tiene que visualizar es el saldo actual, que de acuerdo a la prueba anterior quedo en \$600.00:

En este Frame de Saldo, el botón continuar debe de cerrar la aplicación, por lo que en el evento del botón solo agregamos la siguiente línea de código: **this.dispose();**

Para una mejor apreciación de la ejecución del proyecto se agrega un video que se puede visualizar en el siguiente link: https://drive.google.com/file/d/1aWTNSZESrtgLPe9iLCLFq_cLp3AY3xw1/view?usp=sharing

CONCLUSIONES

Java, como lenguaje multiplataforma, permite crear aplicaciones que se ejecutan en diferentes sistemas operativos, mientras que NetBeans, como Entorno Desarrollado Integrado (IDE), proporciona un entorno de desarrollo completo y eficiente para Java y otros lenguajes, por tanto podemos concluir que NetBeans y Java son herramientas para la programación de software de gran ventaja y utilidad ya que más allá de que son gratuitos y fácil uso ya que su interfaz es intuitiva ambas herramientas juntas son una buena combinación para aplicaciones a nivel empresarial que requieren de gran fiabilidad y seguridad, aprovechando sus ventajas para crear código modular, re utilizable y fácil de mantener.

Al combinar estas dos herramientas tenemos estas ventajas:

- Productividad: permiten la creación de aplicaciones de alta calidad
- Flexibilidad: soporta múltiples lenguajes.
- Facilidad de uso: el uso de su interfaz es muy intuitiva
- Escalabilidad y seguridad, lo cual es necesario para el desarrollo de diferentes proyectos.

Así mismo en esta actividad, aunque en cierto modo, en menor cantidad de código, estuvimos trabajando con XAMPP que facilita la creación y gestión bases de datos, en este caso MySQL, lo que nos permitió realizar la Base de Datos necesaria para poder estar interactuando con la aplicación desarrollada en NetBeans.

Se agrega dicha actividad a la plataforma de GitHub a través del siguiente link:

<https://github.com/22HADRIA/Lenguajes-de-Programaci-n-IV>

REFERENCIAS

Entorno NetBeans

https://repositorio.konradlorenz.edu.co/micrositios/001-985/entorno_netbeans.html

¿Qué es NetBeans? Ventajas y usos

<https://immune.institute/blog/que-es-netbeans/>

Programación Orientada a Objetos

<https://sites.google.com/site/programacionbasicajava/home/programaci%C3%B3n-orientada-a-objetos>

¿Qué es Netbeans? ¿Crea aplicaciones con Java a la velocidad de la luz!

<https://www.crehana.com/blog/transformacion-digital/que-es-netbeans/>

¿Qué es NetBeans IDE? La forma más inteligente y rápida de programar.

<https://www.oracle.com/application-development/netbeans/>